Call-by-Value solvability, revisited

Beniamino Accattoli, Luca Paolini

Call-by-Value Solvability, Revisited

Beniamino Accattoli[1] and Luca Paolini[2]

[1] INRIA and LIX (École Polytechnique), France
[2] Dipartimento di Informatica, Università degli Studi di Torino, Italy

Abstract. In the call-by-value lambda-calculus solvable terms have been characterised by means of call-by-name reductions, which is disappointing and requires complex reasonings. We introduce the value-substitution lambda-calculus, a simple calculus borrowing ideas from Herbelin and Zimmerman's call-by-value λ_{CBV} calculus and from Accattoli and Kesner's substitution calculus $\lambda_{\mathtt{sub}}$. In this new setting, we characterise solvable terms as those terms having normal form with respect to a suitable restriction of the rewriting relation.

1 Introduction

The most commonly used parameter passing policy for programming languages is call-by-value (CBV). Landin in [15] pioneered a CBV formal evaluation for a lambda-core of ALGOL60 (named ISWIM) via the SECD abstract machine. Ten years later, Plotkin [22] introduced the $\lambda\beta_v$-calculus in order to grasp the CBV paradigm in a pure lambda-calculus setting. The $\lambda\beta_v$-calculus narrows the β-reduction rule by allowing the contraction of a redex $(\lambda x.t)\ u$, only in case u is a *value*, *i.e.* a variable or an abstraction. Unfortunately, the semantics analysis of the $\lambda\beta_v$-calculus has turned out to be more elaborate than the semantic of the classical call-by-name (CBN) λ-calculus.

CBN and CBV solvability. Solvability [8,13,14] is a pervasive notion in the semantic analysis of (CBN) lambda-calculus (see [26,7]). For instance, it underlies the fundamental notions of approximants, Böhm-trees, separability, and sensible λ-theories. A term t is *solvable* if there exists a head context H s.t. $H[t] \to_\beta^* I$, where $I = \lambda x.x$ is the identity. If t is not solvable then it is *unsolvable*. Solvability was first considered in connection with λ-definability of partial recursive functions. It was noted that representing the everywhere undefined function using the set of terms without normal form is not adequate, such a function should rather be associated to unsolvable terms, which form a strict subset of the set of terms without a normal form. Quoting from [26]:

> [...] only those terms without normal forms which are in fact unsolvable can be regarded as being "undefined" (or better now: "totally undefined"); by contrast, all other terms without normal forms are at least partially defined. Essentially the reason is that unsolvability is preserved by application and composition [...] which [...] is not true in general for the property of failing to have a normal form.

T. Schrijvers and P. Thiemann (Eds.): FLOPS 2012, LNCS 7294, pp. 4–16, 2012.

In CBN unsolvable (resp. solvable) terms can be characterized operationally as the terms without (resp. with) a head normal form. A solid theory of CBV is expected to enjoy an operational characterization of solvability, *i.e. a strategy* which terminates if and only if the term is solvable. The idea is that such a strategy gives a notion of evaluation for the represented functions.

A term t is *CBV-solvable* whenever there is a head context H s.t. $H[t] \to_{\beta_v}^* I$. An operational characterization has been provided in [21,23] but, unfortunately, it is obtained through *call-by-name* β-reduction, which is disappointing and not satisfying. The result is improved in [20] where the characterisation is built upon strong normalization via CBN weak[1] reduction. An operational characterisation of solvability gives a way to compute the results of the represented functions. If it is not possible to get an *internal* characterisation, *i.e.* one which uses the rules of the calculus itself, then there is an inherent weakness in the rewriting rules of the calculus. For λ_{β_v} it is indeed the case, let us illustrate the point with an example. Let $\Delta = \lambda x.xx$. There is no *head* context sending (via β_v-reduction) the following term to the identity:

$$t = (\lambda y.\Delta)\ (x\ z)\ \Delta \tag{1}$$

and—as a consequence—t should be *unsolvable* and *divergent* in a good call-by-value calculus, while it is in $\lambda\beta_v$-normal form (!). The weakness of β_v-reduction is a fact widely recognized and accepted, indeed there have been many proposals of alternative CBV calculi [11,12,17,24,9].

The value-substitution $\lambda_{\mathtt{vsub}}$-calculus. In this paper we introduce the value-substitution $\lambda_{\mathtt{vsub}}$-calculus, a simple CBV calculus with two rewriting rules. It extends the syntax of λ-calculus with an explicit substitution constructor $t[u/x]$ (an avatar of `let`-expressions), but these substitutions are just delayed, they are not propagated in a small-step way. Our calculus borrows ideas from two existing calculi, Herbelin and Zimmerman's λ_{CBV}-calculus [11] and Accattoli and Kesner's $\lambda_{\mathtt{sub}}$-calculus [4], as we explain in Section 2. In particular, it is a reformulation *at a distance* [5,4]—*i.e.* without commutative rules—of λ_{CBV}.

We prove that in the value-substitution calculus solvable terms can be characterised *internally* as the terms having normal form with respect to a sub-reduction of the calculus that we call *stratified-weak reduction*. Stratified-weak reduction plays in our characterization the same role of head β-reduction in CBN. The characterisation is obtained in two steps. First, we tackle the weaker notion of *potentially valuable terms*, *i.e.* terms for which there exists a substitution sending them into *values* (note that the identity is a value). Such terms are shown to be exactly those having normal form with respect to weak $\lambda_{\mathtt{vsub}}$-reduction. Second, solvable terms are shown to be sort of *hereditarily* potentially valuable terms.

Behavioural equivalence. The gain in moving from λ_{CBV} to $\lambda_{\mathtt{vsub}}$ is the fact that $\lambda_{\mathtt{vsub}}$ can be extended with a behavioural equivalence $\equiv_{\mathtt{vo}}$, induced by the

[1] Weak β-reduction is the applicative-only closure of the β-rule, *i.e.* it is obtained from usual β-reduction by forbidding reductions under abstractions. In literature, it is sometimes called lazy β-reduction.

absence of commutative rules. The idea is that $\equiv_{\mathtt{vo}}$ relates terms differing only for the position of substitutions but *behaving the same*. Formally, $\equiv_{\mathtt{vo}}$ is a strong bisimulation of $\lambda_{\mathtt{vsub}}$ with itself. The calculus $\lambda_{\mathtt{vsub}}$ modulo $\equiv_{\mathtt{vo}}$ is particularly well-behaved, roughly because strong bisimulations preserve most operational properties. We use $\equiv_{\mathtt{vo}}$ to show that our characterisation of solvability in $\lambda_{\mathtt{vsub}}$ lifts to λ_{CBV}.

The value-substitution calculus can also be related to the call-by-value translation $(\cdot)^v$ of λ-calculus with explicit substitutions into multiplicative and exponential intuitionistic linear logic (IMELL) proof-nets, identified by $(A \Rightarrow B)^v = !(A^v \multimap B^v)$ [10], which actually was our starting point. In particular, the equivalence $\equiv_{\mathtt{vo}}$ relates terms which map to the same proof-net. However, proof-nets do not appear in this paper.

Related work. There exist various alternative approaches to CBV [11,12,17,24]. Unlike $\lambda_{\mathtt{vsub}}$ they all have many rewriting and commutative rules, and for none of them solvability has been studied. Since $\lambda_{\mathtt{vsub}}$ is essentially a refinement of λ_{CBV}, we compare them explicitly in Section 2.1, while we refer to the introduction of [11] for more relations with the literature. Solvability has also been recently studied for some extensions of λ-calculus in [18,25], but both works consider a call-by-name calculus.

Outline. Section 2 introduces the calculus, shows that it is a sub-calculus of λ_{CBV} and provides a simple proof of confluence for $\lambda_{\mathtt{vsub}}$. Section 3 introduces CBV-solvability and explains the problems it poses. Section 4 proves that terms having weak normal form are potentially valuable and that terms having stratified-weak normal form are solvable. Section 5 proves the converse implications. Section 6 introduces the behavioural equivalence and lifts the characterisation of solvability to λ_{CBV}.

Proofs. We omit all proofs, which can be found in [6].

2 The Value-Substitution Calculus

The value-substitution calculus $\lambda_{\mathtt{vsub}}$ is a lambda-calculus with explicit substitutions whose syntax is given by the following grammar:

$$\mathtt{v} ::= x \mid \lambda x.t \qquad\qquad t, s, r ::= \mathtt{v} \mid t\ s \mid t[s/x]$$

where x, y, z are variables, $\mathtt{v}$ is the set of values and $t[s/x]$ denotes an explicit substitution, *i.e.* a syntactical representation of a delayed substitution. In $t[s/x]$, the subterm s is called the *content* of the explicit substitution. We use $t\{s/x\}$ for the term obtained by the *capture-avoiding* substitution of s to each occurrence of x in t. There are two kinds of binder: $\lambda x.t$ and $t[u/x]$, both binding x in t. All terms are considered up to α-conversion. Contexts are defined via the grammar:

$$\mathtt{C} ::= [\cdot] \mid \lambda x.\mathtt{C} \mid \mathtt{C}\ t \mid t\ \mathtt{C} \mid \mathtt{C}[t/x] \mid t[\mathtt{C}/x]$$

where $[\cdot]$ is a fresh constants. We use $\mathtt{C}[t]$ for the term obtained by the *capture-allowing* substitution of t to $[\cdot]$ in $\mathtt{C}$ and $\mathtt{L}$ for a (possibly empty) lists

$[t_1/x_1]\dots[t_k/x_k]$ of explicit substitutions. The value-substitution calculus is endowed with two rewriting rules ($\mathtt{dB}$ for $\mathtt{B}$ *at a distance*[2] and $\mathtt{vs}$ for *value-substitution*):

$$(\lambda x.t)\mathtt{L}\ s \mapsto_{\mathtt{dB}} t[s/x]\mathtt{L} \qquad\qquad t[\mathtt{v}\mathtt{L}/x] \mapsto_{\mathtt{vs}} t\{\mathtt{v}/x\}\mathtt{L}$$

We use $\to_{\mathtt{dB}}, \to_{\mathtt{vs}}$ and $\to_{\lambda_{\mathtt{vsub}}}$ for the closure by all contexts of $\mapsto_{\mathtt{dB}}$, $\mapsto_{\mathtt{vs}}$ and $\mapsto_{\mathtt{dB}} \cup \mapsto_{\mathtt{vs}}$, respectively. Some comments on the rewriting rules are in order. The rule $\to_{\mathtt{dB}}$ does *not* require the argument of the redex to be a value. The rule $\to_{\mathtt{vs}}$ instead can fire only when the content of the explicit substitution is of the form $\mathtt{v}\mathtt{L}$ (*i.e.* a value followed by a list of substitutions), generalizing the usual requirement of being a value. Note that the unsolvable term t in (1) (page 5) diverges in $\lambda_{\mathtt{vsub}}$:

$$\begin{array}{l} t = (\lambda y.\Delta)\ (x\ z)\ \Delta \quad \to_{\mathtt{dB}} \Delta[x\ z/y]\ \Delta \to_{\mathtt{dB}} \\ \qquad x'\ x'[\Delta/x'][x\ z/y] \to_{\mathtt{vs}} \Delta\ \Delta[x\ z/y] \to_{\mathtt{dB}} \dots \end{array} \tag{2}$$

The embedding of Plotkin's $\lambda\beta_v$-calculus into $\lambda_{\mathtt{vsub}}$ is simply given by the identity, and each β_v-reduction step $(\lambda x.t)\ \mathtt{v} \to_{\beta_v} t\{\mathtt{v}/x\}$ factors in $\lambda_{\mathtt{vsub}}$ as $(\lambda x.t)\ \mathtt{v} \to_{\mathtt{dB}} t[\mathtt{v}/x] \to_{\mathtt{vs}} t\{\mathtt{v}/x\}$, as in the call-by-value calculi of [11,17]. The presence of the list of substitutions $\mathtt{L}$ may not seem necessary, but it is in fact the key to avoid commutation rules, as we explain in the next subsection. The following immediate lemma will be used implicitly throughout the paper.

Lemma 1. *If* $\mathtt{v}_0, \mathtt{v}_1 \in \mathit{Val}$ *then* $\mathtt{v}_0\{\mathtt{v}_1/x\} \in \mathit{Val}$. *Moreover,* $\mathtt{v}_0 \to_{\lambda_{\mathtt{vsub}}} \mathtt{v}_0'$ *implies* $\mathtt{v}_0' \in \mathit{Val}$.

2.1 Relation with Herbelin's and Zimmerman's λ_{CBV}

The calculus we introduced borrows ideas from two already existing calculi, Herbelin and Zimmerman's λ_{CBV}[11] and Accattoli and Kesner's $\lambda_{\mathtt{sub}}$ [4]. Both calculi extend the syntax of λ-calculus: the former uses a $\mathtt{let}\ x = u\ \mathtt{in}\ t$ construct, while the latter uses an explicit substitution construct $t[u/x]$. The two construct are in fact equivalent: we present both calculi with explicit substitutions, since $\mathtt{let}$ is quite verbose and easily gives rise to big terms.

A key feature of λ_{CBV} is that the CBV restriction on redexes is imposed on explicit substitutions and not on β-redexes. The rewriting rules of λ_{CBV}—omitting the observational ones—follow.

Operational rules			Structural rules		
$(\lambda x.t)\ s$	$\Rightarrow$	$t[s/x]$	$t[u/x]\ s$	$\to_{let_{app}}$	$(t\ s)[u/x]$
$t[\mathtt{v}/x]$	$\to_{let_v}$	$t\{\mathtt{v}/x\}$	$t[s[u/y]/x]$	$\to_{let_{let}}$	$t[s/x][u/y]$

Structural rules commute explicit substitutions to enable hidden operational redexes. For instance, $(\lambda x.t)[u/y]\ s$ becomes a $\Rightarrow$-redex only after the structural

[2] $\mathtt{B}$ is often used for the rule $(\lambda x.t)\ s \to t[s/x]$.

step $(\lambda x.t)[u/y]\ s \to_{let_{app}} ((\lambda x.t)\ s)[u/y]$. Similarly, $t[\mathtt{v}[u/y]/x]$ becomes a $\to_{let_v}$-redex only after a $\to_{let_{let}}$-step.

The substitution calculus $\lambda_{\mathtt{sub}}$ is a CBN calculus with explicit substitutions designed to reflect reductions in $\lambda\mathtt{j}$-dags [3] and pure proof-nets [2]. It has two rewriting rules:

$$(\lambda x.t)\mathtt{L}\ s \to_{\mathtt{dB}} t[s/x]\mathtt{L} \qquad\qquad t[u/x] \to_{\mathtt{s}} t\{u/x\}$$

where $\mathtt{L}$ is a list of substitutions, like in $\lambda_{\mathtt{vsub}}$. The main feature of $\lambda_{\mathtt{sub}}$ is *distance*, *i.e.* the absence of commutative rules for substitutions: in $\to_{\mathtt{dB}}$ the function $\lambda x.t$ and the argument s can interact even if there is $\mathtt{L}$ between them. This is motivated by the close relation between $\lambda_{\mathtt{sub}}$ and graphical formalisms as (Pure) Proof-Nets or $\lambda\mathtt{j}$-dags, see [3,2]. The value-substitution calculus is a reformulation *at a distance* of λ_{CBV}, making the structural rules superfluous.

The rules of $\lambda_{\mathtt{vsub}}$ are sort of *macro-rules* of λ_{CBV}:

$$\begin{array}{ccccc} (\lambda x.t)\mathtt{L}\ s & \to^*_{let_{app}} & ((\lambda x.t)\ s)\mathtt{L} & \Rightarrow & t[s/x]\mathtt{L} \\ t[\mathtt{v}\mathtt{L}/x] & \to^*_{let_{let}} & t[\mathtt{v}/x]\mathtt{L} & \to_{let_v} & t\{\mathtt{v}/x\}\mathtt{L} \end{array}$$

that provide a straightforward simulation of $\lambda_{\mathtt{vsub}}$ into λ_{CBV}.

Proposition 1. *$\to_{\lambda_{\mathtt{vsub}}} \subseteq \to^+_{CBV}$, and so the equational theory of $\lambda_{\mathtt{vsub}}$ is contained in the theory of λ_{CBV}.*

Akin to other CBV calculi, λ_{CBV} equates more than Plotkin's calculus. Indeed, the two terms:

$$(\lambda x.\lambda x'.t)\ (y\ y')\ (z\ z') \qquad\qquad (\lambda x.((\lambda x'.t)\ (z\ z')))\ (y\ y') \tag{3}$$

are *not* β_v-interconvertible, while in λ_{CBV} both reduce to $t[z\ z'/x'][y\ y'/x]$. In Section 6 we show that in a sense $\lambda_{\mathtt{vsub}}$ (strictly) contains the equational theory of λ_{CBV}, despite the fact that $\lambda_{\mathtt{vsub}}$ is a subcalculus of λ_{CBV}.

2.2 Confluence

The proof of confluence is particularly simple. It is based on the following well-known lemma (used, for instance, to prove confluence of the $\lambda\eta$-calculus).

Lemma 2 (Hindley-Rosen, [7], Proposition 3.3.5.(ii), page 64). *Let $\to_1$ and $\to_2$ be two rewriting relations on a set X. If they are both confluent and they commute,* i.e. *if $t \to^*_1 u_1$ and $t \to^*_2 u_2$ then there exists s such that $u_1 \to^*_2 s$ and $u_2 \to^*_1 s$, then $\to_1 \cup \to_2$ is confluent.*

The idea is to take $\to_1 = \to_{\mathtt{dB}}$, $\to_2 = \to_{\mathtt{vs}}$ and $\to_1 \cup \to_2 = \to_{\lambda_{\mathtt{vsub}}}$ and prove the hypothesis of the lemma. Confluence of $\to_{\mathtt{dB}}$ and $\to_{\mathtt{vs}}$ follows from their respective local confluence, Newman's Lemma and the fact that they are strongly normalising (separately).

Lemma 3. *$\to_{\mathtt{dB}}$ and $\to_{\mathtt{vs}}$ are both confluent and strongly normalising reductions.*

Commutation of $\to_{\mathtt{dB}}$ and $\to_{\mathtt{vs}}$ follows by an easy and standard argument based on the particular shape of their local commutation diagram.

Lemma 4. *1. $\to_{\mathtt{vs}}$ **and** $\to_{\mathtt{dB}}$ **locally commute**: If $t \to_{\mathtt{vs}} u_1$ and $t \to_{\mathtt{dB}} u_2$ then there is s s.t. $u_2 \to_{\mathtt{vs}} s$ and $u_1 \to^*_{\mathtt{dB}} s$.*
*2. $\to_{\mathtt{vs}}$ **and** $\to_{\mathtt{dB}}$ **commute**: If $t \to^*_{\mathtt{vs}} u_1$ and $t \to^*_{\mathtt{dB}} u_2$ then there is s s.t. $u_2 \to^*_{\mathtt{vs}} s$ and $u_1 \to^*_{\mathtt{dB}} s$.*

Thus Lemma 2 gets:

Corollary 1. *$\to_{\lambda_{\mathtt{vsub}}}$ is confluent.*

3 Call-by-Value Solvability

First of all, let us recall the definition of solvability.

Definition 1 (Solvable Terms). *A term t is **solvable** if there exist terms $u_1, \dots, u_k$ and variables $x_1, \dots, x_h$, with $h, k \geq 0$, such that $(\lambda x_1 \dots x_h.t)\ u_1 \dots u_k \to^*_{\lambda_{\mathtt{vsub}}} I$, where I is the identity.*
*We call $(\lambda x_1 \dots \lambda x_h.[\cdot])\ u_1 \dots u_k$ a **head context**.*

Let Δ be $\lambda x.xx$. With respect to solvability the difference between CBN and CBV is well represented by the term $t = I[\Delta\ \Delta/x]$. The subterm $\Delta\ \Delta$ is a typical example of unsolvable term. In CBN one has that $t \to I$ by simply erasing the substitution, and thus t is CBN solvable. In a CBV setting, instead, the substitution is blocked, because $\Delta\ \Delta$ is not a value, nor it can be reduced to a value. Even worse, no plugging of t in a head context can turn $\Delta\Delta$ in a value. Thus, there is no head context sending t on the identity, and t is CBV-unsolvable. We need to find a notion of reduction for which t diverges.

To understand the difficulty is useful to recall the structure of the proof of the characterisation of CBN-solvability:

1. *t has head normal form $\Rightarrow$ t is solvable*: this direction is proved by induction on the structure of t building a special head context H which *erases* all non-head subterms of t and produces the identity;
2. *t is solvable $\Rightarrow$ t has head normal form*: it is a corollary of the standardisation theorem, if $H[t]$ reduces to the identity then it does so by head reduction, and the fact that extraction of t from $H[t]$ preserves head normalisability.

We adapt the same pattern of the CBN approach, defining a new form of reduction (to be introduced in a while). The main difference is that in CBV not every non-head subterm can be erased, only values. Thus the proof of the first step is more involved. The head context transforming a solvable term into the identity needs to provide appropriate substitutions turning the content of explicit substitutions into values. So it is mandatory to first characterise *potentially valuable* terms.

Definition 2 (Potentially valuable term, [21]). *A term t is* **potentially valuable** *whenever there are terms $u_1, \dots, u_n$, variables $x_1 \dots x_n$ and a value v such that $t\{u_1/x_1\} \dots \{u_n/x_n\} \to^*_{\lambda_{\mathtt{vsub}}} \mathsf{v}$*[3].

For example, values, and terms which reduce to values, are potentially valuables. A potentially valuable term which does not reduce to a value is $x\ y$ (consider $\{I/x\}$), while $\Delta\Delta$ is not potentially valuable. We show that a term is potentially valuable iff it has a weak normal form, and then we will build the characterisation of solvability on top of this one.

Weak contexts are contexts whose hole is not under an abstraction:

$$\mathtt{W} ::= [\cdot] \mid \mathtt{W}\ t \mid t\ \mathtt{W} \mid \mathtt{W}[t/x] \mid t[\mathtt{W}/x]$$

The reduction $\to_{\mathtt{w}}$ is the closure by *weak* contexts of $\mapsto_{\mathtt{dB}} \cup \mapsto_{\mathtt{vs}}$. We note $\to_{\neg\mathtt{w}}$ the complement of $\to_{\mathtt{w}}$, defined as $\to_{\lambda_{\mathtt{vsub}}} \setminus \to_{\mathtt{w}}$, which is the reduction which reduces redexes under at least one λ.

To catch solvability we extend weak reduction as follows. A **stratified-weak context** $\mathtt{SW}$ is defined as:

$$\mathtt{SW} ::= \mathtt{W} \mid \lambda x.\mathtt{SW} \mid \mathtt{SW}\ t \mid \mathtt{SW}[t/x]$$

The reduction $\to_{\mathtt{sw}}$ is the closure by *stratified-weak* contexts of $\mapsto_{\mathtt{dB}} \cup \mapsto_{\mathtt{vs}}$.

Weak contexts are widely used in literature, while stratified-weak contexts are an adptation of the ahead-strategy defined in [21]. They extend weak contexts allowing weak reductions under abstractions in head position, which have the important property that cannot be duplicated nor erased. Note that the diverging reduction (2) (page 7) of the unsolvable term of the introduction is a weak (and thus a stratified-weak) reduction.

4 Terms Having Stratified-weak Normal Form Are Solvable

Let us sketch the organization of this section.

1. We prove that terms having a weak normal form are potentially valuable, by:
   (a) characterising weak normal forms explicitly;
   (b) proving that weak normal forms are potentially valuable;
   (c) proving that terms having weak normal form are potentially valuable.
2. We prove that terms having a stratified-weak normal form are solvable, by:
   (a) characterising stratified-weak normal forms explicitly;
   (b) proving that stratified-weak normal forms are solvable;
   (c) proving that terms having stratified-weak normal form are solvable.

[3] Potentially valuable terms can be defined via head-contexts, as for solvable terms, but our definition simplifies some technical points.

The characterization of weak normal forms, noted $\mathsf{w}_{\mathtt{nf}}$, uses an auxiliary syntactic category $\mathsf{w}^{\#}_{\mathtt{nf}}$ for terms which have not the form vL. The idea is that a substitution $t[u/x]$ can be reduced by $\mapsto_{\mathtt{vs}}$ iff $u \notin \mathsf{w}^{\#}_{\mathtt{nf}}$.

Lemma 5 (Weak normal forms). *Let $t \in \lambda_{\mathtt{vsub}}$ and consider the following grammar:*

$$\begin{array}{lll} \mathsf{w}_{\mathtt{nf}} ::= x \mid \lambda x.t \mid \mathsf{w}^{\#}_{\mathtt{nf}} \mid \mathsf{w}_{\mathtt{nf}}[\mathsf{w}^{\#}_{\mathtt{nf}}/x] & & \textit{(weak n.f.)} \\ \mathsf{w}^{\#}_{\mathtt{nf}} ::= x[\mathsf{w}^{\#}_{\mathtt{nf}}/x_1]\dots[\mathsf{w}^{\#}_{\mathtt{nf}}/x_n]\ \mathsf{w}_{\mathtt{nf}} \mid \mathsf{w}^{\#}_{\mathtt{nf}}\mathsf{w}_{\mathtt{nf}} \mid \mathsf{w}^{\#}_{\mathtt{nf}}[\mathsf{w}^{\#}_{\mathtt{nf}}/x] & & \textit{(\#-weak n.f.)} \end{array}$$

with x possibly among $x_1, \dots, x_n$ and $n \geq 0$. Then:

1. *t is in $\to_{\mathsf{w}}$-normal form iff $t \in \mathsf{w}_{\mathtt{nf}}$.*
2. *t is in $\to_{\mathsf{w}}$-normal form and not of the form vL iff $t \in \mathsf{w}^{\#}_{\mathtt{nf}}$ where $\mathtt{L}$ is a list of substitutions of #-weak normal forms.*

In λ_{CBV}, thanks to the structural rules, weak normal forms are simpler, they have either the shape $xt_1...t_n\mathtt{L}$ or the shape $(\lambda x.t)\mathtt{L}$, where t_i are weak normal forms and $\mathtt{L}$ is a list of explicit substitutions having as content terms in $\mathsf{w}^{\#}_{\mathtt{nf}}$. However, using λ_{CBV} would not get rid of $\mathsf{w}^{\#}_{\mathtt{nf}}$ and the operational study in the paper would be more complex, since λ_{CBV} has more rules. In Section 6 we will show how to characterise solvability in λ_{CBV}, by exploiting the characterisation in $\lambda_{\mathtt{vsub}}$.

We need some meta-notations. If $n \in \mathbb{N}$ then o^n is the term $\lambda x_0...x_n.x_n$ (*i.e.* $\mathsf{o}^0 = \lambda x_0.x_0 = I$ and $\mathsf{o}^{i+1} = \lambda x_{i+1}.\mathsf{o}^i$), moreover we use the notation $\mathsf{o}^{\geq n}$ to denote a term of the shape $\lambda x_0...x_k.x_k$ where $k \geq n$. Clearly, a term $\mathsf{o}^{\geq n}$ can be noted $\mathsf{o}^{\geq n-1}$, loosing information. Let t be a term, we note $t^{[n]}$ a term of the form $t\{\mathsf{o}^{\geq n}/x_1, \dots, \mathsf{o}^{\geq n}/x_k\}$ with $\mathrm{FV}(t) = \{x_1, \dots, x_k\}$. Note that o^n is a closed value and $t^{[n]}$ is a closed term, for all $n \in \mathbb{N}$. The following lemma is one of the key points of the characterisation. Its proof is delicate, technical and non-trivial.

Lemma 6. *Let $t \in \lambda_{\mathtt{vsub}}$.*

1. *If $t \in \mathsf{w}^{\#}_{\mathtt{nf}}$ then exists $h \in \mathbb{N}$ s.t., for all $j \in \mathbb{N}$, $t^{[h+j]} \to^*_{\lambda_{\mathtt{vsub}}} \mathsf{o}^{\geq j}$.*
2. *If $t \in \mathsf{w}_{\mathtt{nf}}$ then exists $h \in \mathbb{N}$ s.t. for all $j \in \mathbb{N}$ exists v s.t. $t^{[h+j]} \to^*_{\lambda_{\mathtt{vsub}}} \mathsf{v}$.*

Therefore, t is potentially valuable.

Since Lemma 6 hides many details in notations to simplify the statement, let us spend some words on its first point. The first point says that substituting a family of terms $\mathsf{o}^{\geq h+j}$ (all of them, with at least $h+j$ abstractions) to *all* the free variables of t we can obtain a term $\mathsf{o}^{\geq j}$ (with at least j abstractions).

The following lemma is used to lift the result to terms having weak normal form, and its proof is straightforward.

Lemma 7. *If $t \to_{\mathsf{w}} t'$ then $t\{\mathsf{v}/x\} \to_{\mathsf{w}} t'\{\mathsf{v}/x\}$.*

It is easily seen that Lemmas 6 and 7 imply the following corollary.

Corollary 2. *If t has a $\to_{\mathsf{w}}$-normal form then it is potentially valuable.*

Now we show that $\to_{\mathsf{sw}}$-normalizing terms are solvable. The first step is a characterisation of stratified-weak normal forms.

Lemma 8 (Stratified-weak normal forms). *A term t is in $\to_{\mathsf{sw}}$-normal form if and only if it belongs to the following syntax:*

$$\mathsf{s}_{\mathtt{nf}} ::= x \mid \lambda x.\mathsf{s}_{\mathtt{nf}} \mid \mathsf{w}^{\#}_{\mathtt{nf}} \mid \mathsf{s}_{\mathtt{nf}}[\mathsf{w}^{\#}_{\mathtt{nf}}/x].$$

The second step is that stratified-weak normal forms are solvable.

Lemma 9. *If t is an $\to_{\mathsf{sw}}$-normal form then there exist $h, k \in \mathbb{N}$ such that, for all $j \in \mathbb{N}$,*

$$t^{[h+j]}\underbrace{\mathsf{o}^{\geq h+j}\ldots\mathsf{o}^{\geq h+j}}_{k} \to^{*}_{\lambda_{\mathsf{vsub}}} \mathsf{o}^{\geq j}.$$

Therefore, t is solvable.

The next immediate lemma is used to lift the result to terms having stratified-weak normal forms, *i.e.* to get the third and last step.

Lemma 10. *If $t \to_{\mathsf{sw}} t'$ then $\mathtt{H}[t] \to_{\mathsf{sw}} \mathtt{H}[t']$ for any head context $\mathtt{H}$.*

The characterisation of solvability easily follows.

Corollary 3. *If t has a $\to_{\mathsf{sw}}$-normal form then t is solvable.*

5 Solvable Terms Have Stratified-weak Normal Form

To complete the characterisation of solvability we need to prove that solvable terms have stratified-weak normal form. The proof of this direction relies on the rewriting properties of stratified-weak reduction, in particular a sort of standardisation theorem stating that $\to^{*}_{\lambda_{\mathsf{vsub}}} \subseteq \to^{*}_{\mathsf{sw}} \to^{*}_{\neg\mathsf{sw}}$, where $\to_{\neg\mathsf{sw}}$ is the complement of $\to_{\mathsf{sw}}$ w.r.t. $\to_{\lambda_{\mathsf{vsub}}}$, and the diamond property for $\to_{\mathsf{sw}}$. Similarly, we show that potentially valuable terms have weak normal form.

Lemma 11 (Diamond property). *$\to_{\mathsf{w}}$ (resp. $\to_{\mathsf{sw}}$) enjoys the diamond property,* i.e.*, if $t \to_{\mathsf{w}} u_i$ (resp. $t \to_{\mathsf{sw}} u_i$) and $u_1 \neq u_0$ then there exists s s.t. $u_i \to_{\mathsf{w}} s$ (resp. $u_i \to_{\mathsf{sw}} s$), for $i = 0, 1$.*

The diamond property is an abstract way to say that morally $\to_{\mathsf{sw}}$ and $\to_{\mathsf{w}}$ are deterministic strategies. Indeed, it implies that all maximal reduction sequences from a given term have the same length, which in turn gives the following corollary.

Corollary 4. *A term t has a $\to_{\mathsf{w}}$-normal form (resp. $\to_{\mathsf{sw}}$-normal form) iff t is $\to_{\mathsf{w}}$-strongly normalisable (resp. $\to_{\mathsf{sw}}$-strongly normalisable).*

We also need two factorisation properties [16], simple forms of standardisation, which can also be seen as postponement properties. Let $\to_{\neg \mathsf{w}}$ be the complement of $\to_{\mathsf{w}}$ w.r.t. $\to_{\lambda_{\mathtt{vsub}}}$.

Theorem 1 (Factorisation).

1. $\to^*_{\lambda_{\mathtt{vsub}}} \subseteq \to^*_{\mathsf{w}} \to^*_{\neg \mathsf{w}}$.
2. $\to^*_{\lambda_{\mathtt{vsub}}} \subseteq \to^*_{\mathsf{sw}} \to^*_{\neg \mathsf{sw}}$.

The proofs of the two properties are non-trivial. To avoid annoying repetitions we approached them abstractly. This lead the first author to develop an abstract technique for factorisation theorems in [1], where the two cases of our interest are proved.

Both solvable (resp. potentially valuable) terms have been defined using $\to^*_{\lambda_{\mathtt{vsub}}}$, but thanks to the factorisation theorems we can limit reductions to stratified-weak (resp. weak) redexes.

Corollary 5. *Let $t \in \lambda_{\mathtt{vsub}}$.*

1. *If $t \to^*_{\lambda_{\mathtt{vsub}}} \mathsf{v}$ then there exists a value v' s.t. $t \to^*_{\mathsf{w}} \mathsf{v}' \to^*_{\neg \mathsf{w}} \mathsf{v}$.*
2. *If $t \to^*_{\lambda_{\mathtt{vsub}}} I$ then $t \to^*_{\mathsf{sw}} I$.*

The simplification given by the corollary is a key point in the proof of the next theorem.

Theorem 2. *Let $t \in \lambda_{\mathtt{vsub}}$. If t is solvable then t has a stratified-weak normal form.*

We conclude with the similar result for potential valuability.

Theorem 3. *Let $t \in \lambda_{\mathtt{vsub}}$. If t is potentially valuable then it has a $\to_{\mathsf{w}}$-normal form.*

6 Behavioural Equivalence and Solvability for λ_{CBV}

At first sight there is no way of simulating λ_{CBV} in $\lambda_{\mathtt{vsub}}$, since the structural rules have no counterpart. However, in $\lambda_{\mathtt{vsub}}$ the rules $\to_{let_{app}}$ and $\to_{let_{let}}$ can be recovered: they are just lifted to another, more subtle level.

In $\lambda_{\mathtt{vsub}}$ two terms can have *the same behavior* and differ only for the position of substitutions, which is not relevant because substitutions do not block redexes. This can be formalized in a precise way, using the standard tool for behavioural equivalence: bisimulations.

Definition 3. *Let $\equiv_{\mathtt{vo}}$ be the equivalence defined as the reflexive, symmetric, transitive, and contextual closure of the following relations:*

$$
\begin{array}{lll}
t[s/x][u/y] & \sim_{\mathtt{vo}_1} t[u/y][s/x] & \text{if } x \notin \mathtt{fv}(u) \& y \notin \mathtt{fv}(s) \\
t\ u[s/x] & \sim_{\mathtt{vo}_2} (t\ u)[s/x] & \text{if } x \notin \mathtt{fv}(t) \\
t[s/x]\ u & \sim_{\mathtt{vo}_3} (t\ u)[s/x] & \text{if } x \notin \mathtt{fv}(u) \\
t[s[u/y]/x] & \sim_{\mathtt{vo}_4} t[s/x][u/y] & \text{if } y \notin \mathtt{fv}(t)
\end{array}
$$

Remark that $\equiv_{\mathtt{vo}}$ allows the commutation of explicit substitutions with every constructor except abstractions[4]. Moreover, $\to_{let_{app}}$ and $\to_{let_{let}}$ are particular cases of $\sim_{\mathtt{vo}_3}$ and $\sim_{\mathtt{vo}_4}$, respectively.

Let $(\mathcal{S}, \to_{\mathcal{S}})$ be a reduction systems. A **strong bisimulation** for $(\mathcal{S}, \to_{\mathcal{S}})$ is a *symmetric* relation $\equiv \subseteq \mathcal{S} \times \mathcal{S}$ s.t. $s \equiv t$ implies that if $s \to_{\mathcal{S}} s'$ then there exists t' s.t. $t \to_{\mathcal{S}} t'$ and $s' \equiv t'$, for any pair $s, t \in \mathcal{S}$.

Lemma 12. *$\equiv_{\mathtt{vo}}$ is a strong bisimulation for both $(\lambda_{\mathtt{vsub}}, \to_{\lambda_{\mathtt{vsub}}})$ and $(\lambda_{\mathtt{vsub}}, \to_{\mathtt{sw}})$.*

Actually, $\equiv_{\mathtt{vo}}$ has a stronger property, it induces a bijection of redexes and an isomorphism of reduction graphs, not just a bisimulation. However, such stronger property seems to not be useful here.

Strong bisimulations behave very well with respect to the underlying rewriting system, they preserve most operational properties. We just state some basic facts (whose proofs are easy, see [2], pp. 86-87).

Lemma 13. *Let $(\mathcal{S}, \to_{\mathcal{S}})$ be a reduction system, $\equiv$ a strong bisimulation for it, and define $\to_{\mathcal{S}/\equiv} := \equiv \to_{\mathcal{S}} \equiv$. Then:*

1. *$\equiv$ preserves reduction lengths;*
2. *$\equiv$ can be postponed with respect to $\to_{\mathcal{S}}$,* i.e. *$\to^*_{\mathcal{S}/\equiv} \subseteq \to^*_{\mathcal{S}} \equiv$;*
3. *if $\to_{\mathcal{S}}$ is confluent then $\to_{\mathcal{S}/\equiv}$ is confluent and $\to_{\mathcal{S}}$ is Church-Rosser modulo $\equiv$;*
4. *$\to_{\mathcal{S}/\equiv}$ preserves $\mathcal{S}$-strong normalization.*

Hence $\to_{\lambda_{\mathtt{vsub}}/\equiv_{\mathtt{vo}}}$ enjoys all these properties ($\to_{\lambda_{\mathtt{vsub}}}$ is confluent). Summing up, in order to study $\lambda_{\mathtt{vsub}}$ modulo $\equiv_{\mathtt{vo}}$ it is enough to study $\lambda_{\mathtt{vsub}}$, since all properties of $\lambda_{\mathtt{vsub}}$ lift to $\lambda_{\mathtt{vsub}}$ modulo $\equiv_{\mathtt{vo}}$, with essentially no effort. It can be shown that the CBV translation of $\lambda_{\mathtt{vsub}}$ to proof-nets [10] maps two $\equiv_{\mathtt{vo}}$-equivalent terms to the same proof-net, and thus $\equiv_{\mathtt{vo}}$-equivalent terms can be considered as the *same* computational object.

The following lemma shows that λ_{CBV} is a subcalculus of $\lambda_{\mathtt{vsub}}$ modulo $\equiv_{\mathtt{vo}}$ (Point 3).

Lemma 14. *We have:*

1. *If $t \to_{\lambda_{CBV}} u$ reducing an operational redex then $t \to_{\lambda_{\mathtt{vsub}}} u$.*
2. *If $t \to_{\lambda_{CBV}} u$ reducing a structural redex then $t \equiv_{\mathtt{vo}} u$.*
3. *If $t \to^*_{\lambda_{CBV}} u$ then $t \to^*_{\lambda_{\mathtt{vsub}}} \equiv_{\mathtt{vo}} u$.*
4. *$t \to^*_{\lambda_{\mathtt{vsub}}} I$ iff $t \to^*_{\lambda_{CBV}} I$.*

The calculus $\lambda_{\mathtt{vsub}}$ modulo $\equiv_{\mathtt{vo}}$ equates more than λ_{CBV}. For instance, $x\ x[y\ y/z] \equiv_{\mathtt{vo}} (x\ x)[y\ y/z]$ while the two terms are different λ_{CBV} normal forms. Define a term t is **solvable in** λ_{CBV} if there exists a head context $\mathtt{H}$ s.t. $\mathtt{H}[t] \to^*_{\lambda_{CBV}} I$ (note the use of $\to_{\lambda_{CBV}}$ instead of $\to_{\lambda_{\mathtt{vsub}}}$). Then, Lemma 14.4 states that **a term is solvable in $\lambda_{\mathtt{vsub}}$ iff it is solvable in λ_{CBV}**.

[4] The relation $\equiv_{\mathtt{vo}}$ is the CBV version of the relation $\equiv_{\mathtt{o}}$ for the structural λ-calculus $\lambda\mathtt{j}$ in [5], of which $\lambda_{\mathtt{sub}}$ is a big-step variant.

We now show that our characterisation of solvability lifts to λ_{CBV}. The calculi λ_{vsub} and λ_{CBV} share the same syntax, and therefore the same notions of weak and stratified-weak contexts. By closing the rules of λ_{CBV} by stratified-weak contexts we get stratified-weak reduction for λ_{CBV}, noted $\to_{\mathsf{sw}_{CBV}}$. The following lemma relates $\to_{\mathsf{sw}}$ and $\to_{\mathsf{sw}_{CBV}}$.

Lemma 15. *We have:*

1. *If* $t \to_{\mathsf{sw}} u$ *then* $t \to^{+}_{\mathsf{sw}_{CBV}} u$.
2. *If* $t \to_{\mathsf{sw}_{CBV}} u$ *reducing an operational redex then* $t \to_{\mathsf{sw}} u$.
3. *If* $t \to^{*}_{\mathsf{sw}_{CBV}} u$ *then* $t \to^{*}_{\mathsf{sw}} \equiv_{\mathsf{vo}} u$.

In order to show that our characterisation of solvability lifts to λ_{CBV} we need the following easy property.

Lemma 16. *The structural rules of* λ_{CBV} *are strongly normalising.*

We can finally conclude.

Theorem 4. *We have:*

1. *t has a* $\to_{\mathsf{sw}}$ *normal form iff t has a* $\to_{\mathsf{sw}_{CBV}}$ *normal form.*
2. *t is solvable in* λ_{CBV} *iff t has a* $\to_{\mathsf{sw}_{CBV}}$*-normal form.*

7 Conclusions and Future Work

We presented λ_{vsub}, a new CBV calculus with explicit substitutions, compared it to Herbelin's and Zimmerman's λ_{CBV}, and proved an *internal* operational characterisation of solvable terms, simplifying and improving over previous results on CBV solvability.

We plan to put forward the study of CBV through λ_{vsub}. First goals are to adapt the logical characterization of solvability based on intersection types given in [21,20], and the separability theorem proved in [19]. Simplifications and improvements are expected.

We are also interested in a small-step variant of λ_{vsub} evaluation, in order to study a call-by-value version of head linear reduction and the connection to call-by-value abstract machines.

Acknowledgements. The first author wants to thank Simona Ronchi Della Rocca for inviting him in Turin in march 2011, where this work began.

References

1. Accattoli, B.: An abstract factorisation theorem for explicit substitutions (2011) (accepted at RTA 2012), `https://sites.google.com/site/beniaminoaccattoli/factorisation.pdf`
2. Accattoli, B.: Jumping around the box. Ph.D. Thesis, Università di Roma La Sapienza (2011)
3. Accattoli, B., Guerrini, S.: Jumping Boxes. In: Grädel, E., Kahle, R. (eds.) CSL 2009. LNCS, vol. 5771, pp. 55–70. Springer, Heidelberg (2009)

4. Accattoli, B., Kesner, D.: The Permutative λ-Calculus. In: Bjørner, N., Voronkov, A. (eds.) LPAR-18 2012. LNCS, vol. 7180, pp. 23–36. Springer, Heidelberg (2012)
5. Accattoli, B., Kesner, D.: The Structural λ-Calculus. In: Dawar, A., Veith, H. (eds.) CSL 2010. LNCS, vol. 6247, pp. 381–395. Springer, Heidelberg (2010)
6. Accattoli, B., Paolini, L.: Call-by-value solvability, revisited (ext. version) (2012), https://sites.google.com/site/beniaminoaccattoli/CBV-solvabilitywithproofs.pdf
7. Barendregt, H.P.: The Lambda Calculus – Its Syntax and Semantics, vol. 103. North-Holland (1984)
8. Barendregt, H.: Solvability in lambda-calculi. The Journal of Symbolic Logic 39(2), 372 (1975)
9. Dyckhoff, R., Lengrand, S.: Call-by-value lambda-calculus and ljq. J. Log. Comput. 17(6), 1109–1134 (2007)
10. Fernández, M., Siafakas, N.: Labelled lambda-calculi with explicit copy and erase. In: LINEARITY, pp. 49–64 (2009)
11. Herbelin, H., Zimmermann, S.: An Operational Account of Call-by-Value Minimal and Classical λ-Calculus in "Natural Deduction" Form. In: Curien, P.-L. (ed.) TLCA 2009. LNCS, vol. 5608, pp. 142–156. Springer, Heidelberg (2009)
12. Hofmann, M.: Sound and complete axiomatisations of call-by-value control operators. Mathematical Structures in Computer Science 5, 461–482 (1995)
13. Hyland, J.M.E.: A Survey of Some Useful Partial Order Relations on Terms of the Lambda Calculus. In: Böhm, C. (ed.) λ-Calculus and Computer Science Theory. LNCS, vol. 37, pp. 83–95. Springer, Heidelberg (1975)
14. Klop, J.W.: On Solvability by λ I - Terms. In: Böhm, C. (ed.) λ-Calculus and Computer Science Theory. LNCS, vol. 37, pp. 342–345. Springer, Heidelberg (1975)
15. Landin, P.J.: A correspondence between ALGOL 60 and Church's lambda-notation: Part I and Part II. Communications of the ACM 8(2-3), 89–101, 158–165 (1965)
16. Melliès, P.A.: A Factorisation Theorem in Rewriting Theory. In: Moggi, E., Rosolini, G. (eds.) CTCS 1997. LNCS, vol. 1290, pp. 49–68. Springer, Heidelberg (1997)
17. Moggi, E.: Computational lambda-calculus and monads. In: LICS, pp. 14–23. IEEE Computer Society Press, Piscataway (1989)
18. Pagani, M., Rocca, S.R.D.: Linearity, non-determinism and solvability. Fundam. Inform. 103(1-4), 173–202 (2010)
19. Paolini, L.: Call-by-Value Separability and Computability. In: Restivo, A., Ronchi Della Rocca, S., Roversi, L. (eds.) ICTCS 2001. LNCS, vol. 2202, pp. 74–89. Springer, Heidelberg (2001)
20. Paolini, L., Pimentel, E., Ronchi Della Rocca, S.: Lazy strong normalization. Electr. Notes Theor. Comput. Sci. 136, 103–116 (2005)
21. Paolini, L., Ronchi Della Rocca, S.: Call-by-value solvability. Theoretical Informatics and Applications 33(6), 507–534 (1999)
22. Plotkin, G.D.: Call-by-name, call-by-value and the λ-calculus. Theoretical Computer Science 1, 125–159 (1975)
23. Ronchi Della Rocca, S., Paolini, L.: The Parametric λ-Calculus: a Metamodel for Computation. Texts in Theoretical Computer Science: An EATCS. Springer, Berlin (2004)
24. Sabry, A., Felleisen, M.: Reasoning about programs in continuation-passing style. LISP and Symbolic Computation 6, 289–360 (1993)
25. Saurin, A.: Standardization and Böhm Trees for $\Lambda\mu$-Calculus. In: Blume, M., Kobayashi, N., Vidal, G. (eds.) FLOPS 2010. LNCS, vol. 6009, pp. 134–149. Springer, Heidelberg (2010)
26. Wadsworth, C.P.: The relation between computational and denotational properties for Scott's D_∞-models of the lambda-calculus. SIAM Journal of Computing 5(3), 488–521 (1976)